IN THE CLAIMS:

Claim 1 is amended herein. Claims 2-19 are added. All pending claims and their present status are reproduced below.

1. (Currently amended) A multithreaded computer based system for enabling a command in a first thread to access[[ing]] data in a second thread comprising:

an embedded pipelined processor capable of having a first program thread and a second program thread in an execution pipeline, said first program thread comprising a first set of instructions, said second program thread comprising a second set of instructions, said embedded processor comprising:

a fetch unit for fetching an instruction from program memory;

a decode unit for decoding said fetched [[feteched]] instruction;

an execution unit for executing said decoded instruction[[s]]; and

a write back unit for writing the results of said executed instruction to an identified storage location;

a first set of data storage devices capable of storing a first state of said embedded processor, wherein said first state is the state of the embedded processor during the execution of the first program thread[[, including]]

~~a first control status register for identifying a first target set of data storage devices from which a first source operand of a first fetched instruction is to be retrieved from and for identifying a second target set of data storage devices to which a first result of a first executed instruction is stored, wherein said first and second target set of data storage devices are different~~;

a second set of data storage devices capable of storing a second state of said embedded processor, wherein said second state is the state of the embedded processor during the execution of the second program thread[[, including:]]

~~a second control status register for identifying a third target set of data storage devices from which a second source operand of a second fetched instruction is to be retrieved from and for identifying a fourth target set of data storage devices to which a second result of a second executed instruction is stored, wherein said third and fourth target set of data storage devices are different~~;

wherein at least said first set of data storage devices includes a control status register for identifying a first target set of data storage devices from which a first source operand of a fetched instruction is to be retrieved and for identifying a second target set of data storage devices to which a first result of an executed instruction is to be stored, wherein at least one of said first or said second target set of data storage devices is included in the second set of data storage devices;

a thread scheduler for identifying which of said program threads said embedded processor executes; and

an instruction set including an instruction that overwrites the first control status register when instructions associated with the first set of data storage devices are executed and overwrites the second control status register

when instructions associated with the second set of data storage devices are executed;

wherein said processor switches between said first and second state in a time period between the end of the execution of a first program instruction in the first thread and the beginning of the execution of a second program instruction in the second thread;

wherein said processor switches between said first and second states by changing a state selection register.

2. (New) The multithreaded computer based system of claim 1, wherein the embedded pipelined processor further includes a peripheral block.

3. (New) The multithreaded computer based system of claim 2, wherein the peripheral block is one of a phase locked loop and a watchdog timer.

4. (New) The multithreaded computer based system of claim 1, wherein the embedded pipelined processor further includes an internal memory unit comprising a flash memory with a shadow static memory.

5. (New) A method of executing instructions in a multithread computer based system having at least a first thread associated with a first context including a set of context registers, the method comprising the steps of:

selecting the first thread associated with the first context;

fetching a first instruction of the first thread which indicates source data registers associated with operands, each operand associated with a context comprising data registers;

decoding the instruction to determine the context and the source data register associated with a first operand;

executing the instruction on the first operand to produce a result; and

storing the result in a destination data register.

6. (New) The method of claim 5, wherein the decoding further comprises decoding the instruction to determine the context and the source data register associated with a second operand, the context associated with the first operand being the first context and the context associated with the second operand being a second context different from the first context.

7. (New) The method of claim 5, wherein the destination data register is part of a second set of context registers of a second thread different from the first thread.

8. (New) The method of claim 7, wherein the decoding step further comprises determining a context of the destination data register for storing the result.

9. (New) The method of claim 5, wherein the executing includes modifying a control and status register to indicate the context of the first operand being different than the first context.

10. (New) The method of claim 9, wherein the executing further includes modifying the control and status register to indicate a context of the destination data register being different than the first context.

11. (New) An apparatus for executing instructions in a multithread computer based system having at least a first thread associated with a first context comprising a set of context registers, the apparatus comprising:

means for selecting the first thread associated with the first context;

means for fetching a first instruction of the first thread which indicates source data registers associated with operands, each operand associated with a context comprising data registers;

means for decoding the instruction to determine the context and the source data register associated with a first operand;

means for executing the instruction on the first operand to produce a result; and

means for storing the result in a destination data register.

12. (New) A multithread embedded processing system wherein cross-thread access is enabled between multiple pipelined threads, the system comprising a processor and a control and status register, the control and status register having:

a set of source thread selection bits for indicating to the processor a source context of a source thread from which source operands are obtained to be used in a currently executed thread; and

a set of destination thread selection bits for indicating to the processor a destination context of a destination thread to which execution results of the currently executed thread are written.

13. (New) The multithread embedded processing system of claim 12, wherein the control and status register further comprises an enable source override bit for the

processor to select between the context of the currently executed thread and the source context indicated in the source thread selection bits.

14. (New) The multithread embedded processing system of claim 12, wherein the control and status register further comprises an enable destination override bit for the processor to select between the context of the currently executed thread and the destination context indicated in the destination thread selection bits.

15. (New) The multithread embedded processing system of claim 12, wherein the multiple pipelined threads comprise:

a hardware thread having a context, the hardware thread communicatively coupled to hardware inputs and outputs for executing tasks; and

a supervisory thread for controlling the multiple pipelined threads using the control and status register, wherein the controlling occurs one thread at a time and includes suspending the one thread, reading the one thread's state, saving the one thread's state, modifying the one thread's state by writing to the one thread's context, and resuming the one thread, thereby changing the tasks being executed in the one thread.

16. (New) The multithread embedded processing system of claim 12, wherein the control and status register is modified by executing a set control and status register instruction that enables a second thread to overwrite the contents of the control and status register of the currently executed thread.

17. (New) The multithread embedded processing system of claim 12, further comprising a plurality of registers shared by the multiple pipelined threads.

18. (New) The multithread embedded processing system of claim 12, wherein each pipelined thread has an associated set of context registers for storing the thread's state, each set of context registers comprising:

a set of general-purpose registers coupled to the processor for executing instructions;

a set of address registers coupled to the processor for functioning as pointers to memory locations;

a stack pointer register coupled to the processor for storing a stack offset;

a high multiple-accumulate result storage register (MAC_HI) coupled to the processor for storing a first set of bits of a multiple-accumulate result;

a low multiple-accumulate result storage register (MAC_LO) coupled to the processor for storing a second set of bits of the multiple-accumulate result;

a rounded and clipped multiple accumulate result storage register (MAC_RC16) coupled to the processor for creating a digital signal of lower precision than a signal provided by the MAC_HI and MAC_LO registers;

a source register coupled to the processor for providing an additional operand in instructions requiring more than 2 operands; and

a context counter register coupled to the processor for maintaining a count of executed instructions in the thread with which the context is associated.

19. (New) The multithread embedded processing system of claim 18, wherein each register is a 32 bit register.